BE 2018 - Gestion d'un fichier de données

```
clearvars;
```

1- Chargement et prétraitement des données

1a- Reconnaître et charger différents types de fichiers

Q1. Lister et classer en grandes familles les formats de fichiers de données tabulaires ?

- Fichier texte
- Fichier CSV : Common Separeted Values. C'est aussi un fichier texte ! Présence d'un séparateur dans le fichier texte (; , ou un espace). Cela reste mal défini !
- Fichier excel (xls, xlsx) (ou open office)
- Fichier Matlab (.mat)
-

Q2. Lister les différences entre une table et un tableau classique de Matlab (‘array’, matrice…) ?

- Structures propres au soft : matrices, structure, table, ...

```
%help table;
```

Récupération des données du fichier d'archives météo

```
meteorennes=readtable('meteo_rennes_2015.txt');
```

Et on renomme les champs qui nous intéressent :

```
meteorennes.Properties.VariableNames(1)={'Year'};
meteorennes.Properties.VariableNames(2)={'Month'};
meteorennes.Properties.VariableNames(3)={'Day'};
meteorennes.Properties.VariableNames(4)={'Hour'};
meteorennes.Properties.VariableNames(5)={'Temperature'};

temperature=meteorennes.Temperature/10;
```

1b- Données manquantes ou aberrantes

On trace de manière brute la météo issue du fichier de Rennes.

```
plot(temperature)
```

On cherche les limites :

```
maxtemp=max(temperature)
```

```
maxtemp = 35.1000
```

```
mintemps=min(temperature)
```

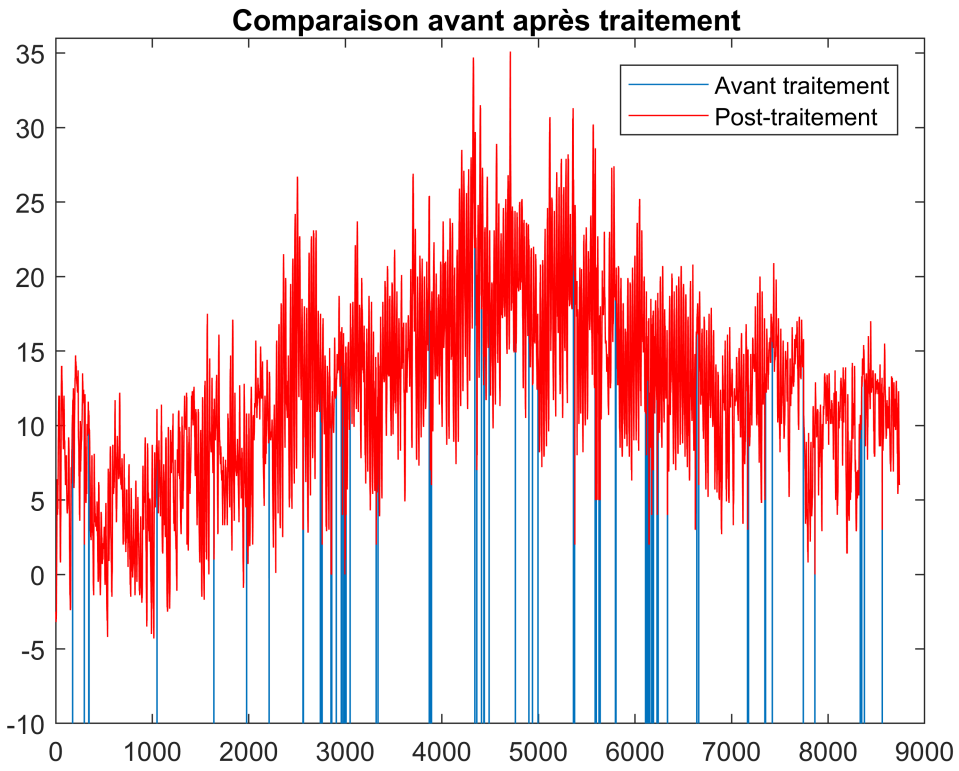
```
mintemps = -999.9000
```

```
ylim([-10 36]) % mise à jour de l'axe des ordonnées
```

On remarque la présence de données aberrantes (-999.9)

On propose alors de les compter, et de les éliminer. Le choix qui est fait ici est de fixer les valeurs aberrantes à la dernière valeur correcte.

```
n_temp=size(temperature,1);
temperature_traite=temperature;
last_temp=temperature(1);
compteur=0;
for i=2:n_temp
    if temperature(i)<-100
        temperature_traite(i)=last_temp; %on affecte la derniere valeur correcte
        compteur=compteur+1;
    else
        last_temp=temperature(i);
    end
end
hold on
plot(temperature_traite,'r')
title('Comparaison avant après traitement ')
legend('Avant traitement','Post-traitement')
```



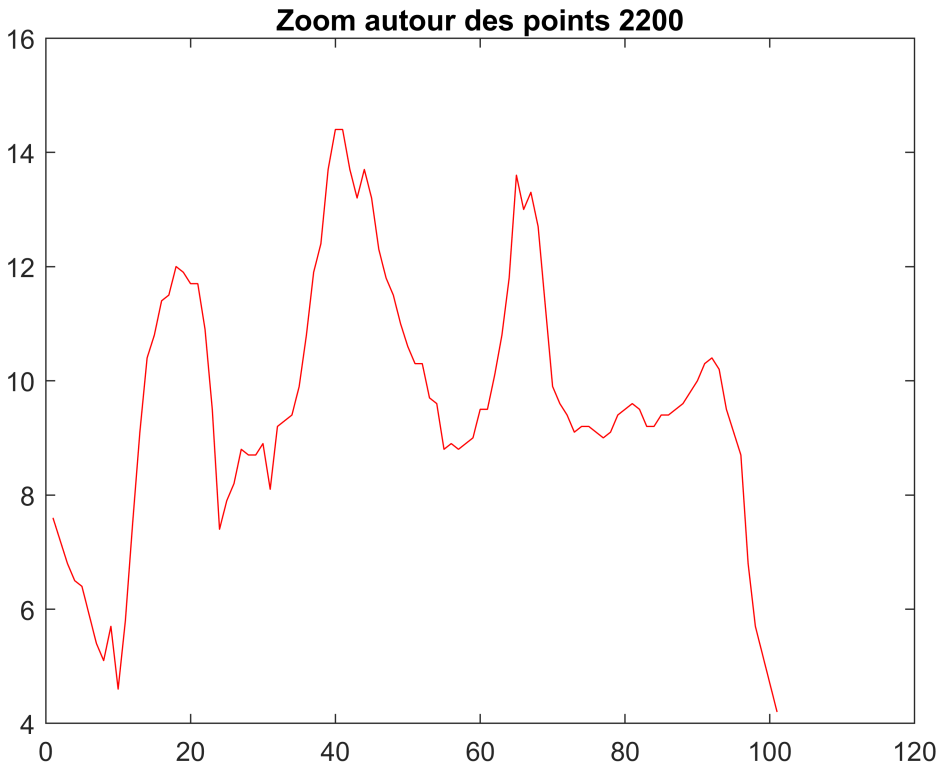
Le nombre de données aberrantes est de :

```
compteur
```

```
compteur = 105
```

On nous propose de faire un zoom vers les valeurs autour de 2200 :

```
figure
plot(temperature_traite(2150:2250),'r')
title('Zoom autour des points 2200')
```



Il paraît qu'il manque des données, effectivement, ça ne se voit pas avec le traitement choisi.

Traitement des valeurs - alternative Pierre Haessig

Pour calculer le nombre de données manquantes

```
temperature_bis=temperature;
manquants=(temperature_bis<-500); % génère un booléen !
nombredemanquants=sum(manquants)
```

```
nombredemanquants = 105
```

```
presents=~manquants;
moyenne=presents'*temperature/(n_temp-nombredemanquants)
```

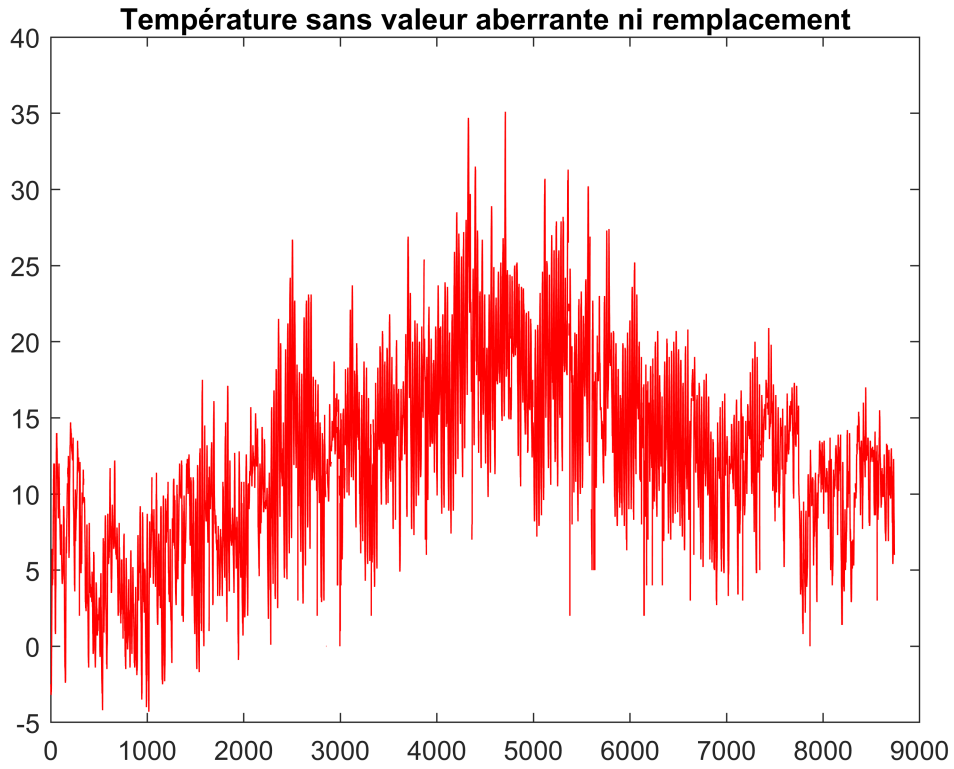
```
moyenne = 12.2487
```

```
% ou la fonction Matlab qui va bien :
temperature_bis(manquants)=missing;
moyennebis=nanmean(temperature_bis)
```

```
moyennebis = 12.2487
```

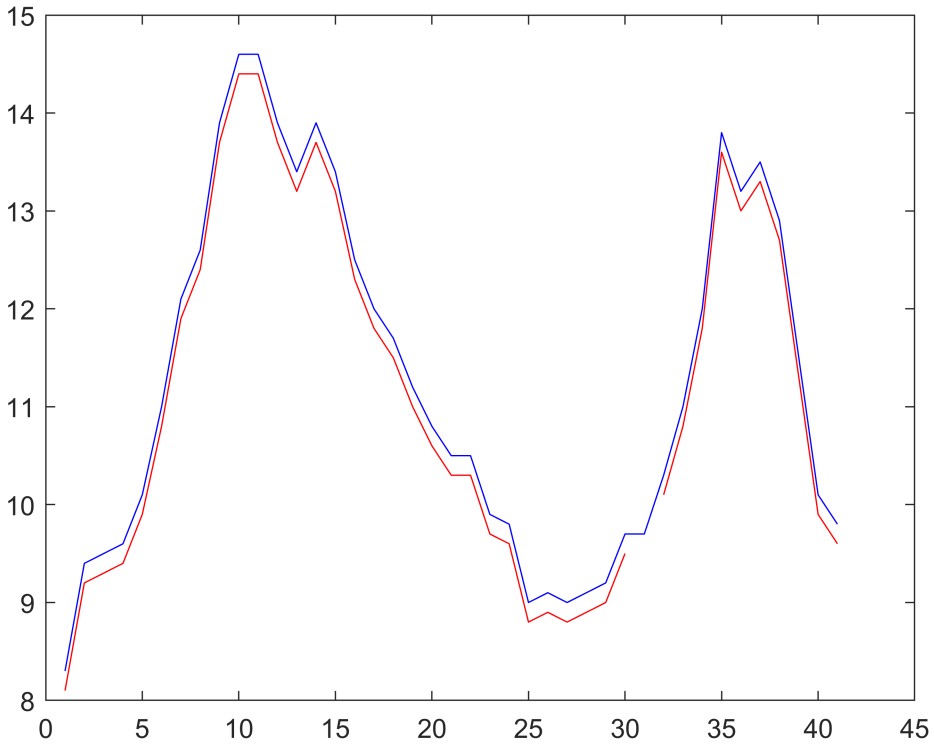
```
figure
```

```
plot(temperature_bis,'r')
title('Température sans valeur aberrante ni remplacement')
```



On peut maintenant comparer au Zoom précédent :

```
figure
plot(temperature_traite(2180:2220)+0.2,'b')
hold on
plot(temperature_bis(2180:2220),'r')
```

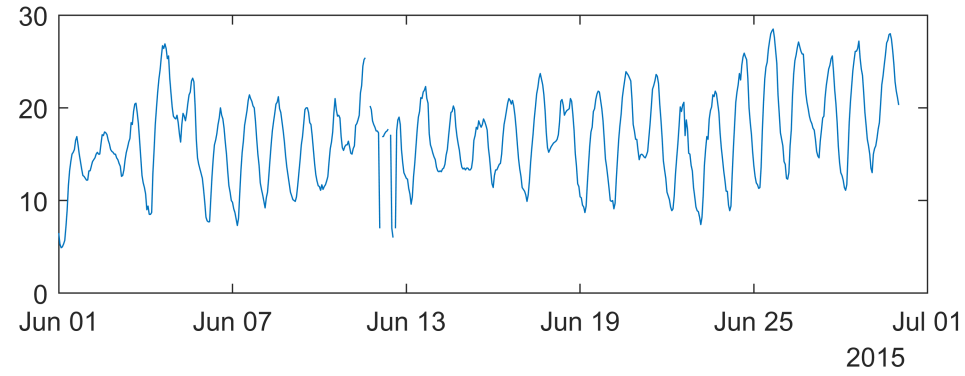


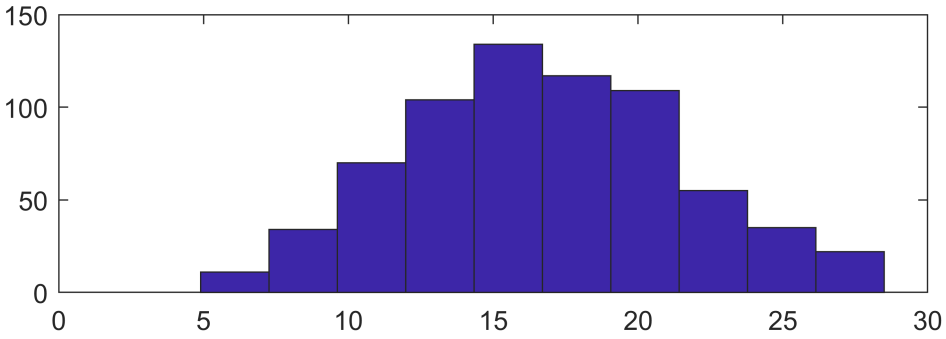
3- Dates : parsing (décodage) et représentation

```
t = datetime(meteorennes.Year,meteorennes.Month,meteorennes.Day,meteorennes.Hour,0,0);

temperature_TT=array2timetable(temperature_bis,'RowTimes',t);
TR=timerange('2015-06-01','2015-06-30','closed')
```

```
TR =
 timetable timerange subscript:

  Select timetable rows with times in the closed interval:
  [01-Jun-2015 00:00:00, 30-Jun-2015 00:00:00]

  See Select Timetable Data by Row Time and Variable Type.
```

```
Temp_juin=temperature_TT(TR,:);

figure
subplot(2,1,1)
plot(Temp_juin.Time,Temp_juin.temperature_bis)
subplot(2,1,2)
hist(Temp_juin.temperature_bis)
```





Traitement des autres données (CO2 - présence)

```
Presence=readtable('fiche_presence.csv');
Capteur=readtable('log-20150309-171821.csv');

class(Capteur.time)
```

```
ans =
'cell'
```

```
tdate = datetime(Capteur.time,'InputFormat','HH:mm:ss.SSS');
tdate.Year=Capteur.date.Year;
tdate.Month=Capteur.date.Month;
tdate.Day=Capteur.date.Day;
```

On a donc récupéré les heures. Il reste maintenant à trier les données

```
Capteur_TT=table2timetable(Capteur,'RowTimes',tdate);
```

```
type=categorical(Capteur_TT.type);
co2= type == 'carbon dioxide';
```

```
CapteurCO2=Capteur_TT(co2,:);
tdateCO2=tdate(co2,:);

figure
plot(CapteurCO2.Time,CapteurCO2.value)
title('Taux de CO2')
```